PAT-NO: JP402238558A

DOCUMENT-IDENTIFIER: JP 02238558 A

TITLE: BOOT SYSTEM FOR PARALLEL COMPUTER

PUBN-DATE: September 20, 1990

INVENTOR-INFORMATION:
NAME
SHIMIZU, TOSHIYUKI
ISHIHATA, HIROAKI

ASSIGNEE-INFORMATION:

| NAME | COUNTRY |
|---|---|
| FUJITSU LTD | N/A |


APPL-NO: JP01060060

APPL-DATE: March 13, 1989

INT-CL (IPC): G06F015/16, G06F009/445 , G06F013/00

ABSTRACT:

PURPOSE: To increase the number of processor elements by executing successively the boot programs received from a communication port via a CPU at the side of each processor element after a host computer detects that all processor elements has accesses to the communication port.

CONSTITUTION: In a reset state an address decoder 4c decodes the address data received from a CPU 4a and has an access to a communication port 4b. A host computer 1 sends a boot program to each processor element PE 4 when the computer 1 detects that all processors PE 4 has accesses to the port 4b. Then

each PE 4 carries out successively the boot programs received from the port 4b
via the CPU 4a. Thus the computer 1 gives a program to each PE 4 in an initial
program load IPL state. As a result, each PE requires no ROM storing an IPL
program and the number of elements PE is increased.

⑲日本国特許庁（JP）　⑪特許出願公開

# ⑫公開特許公報（A）　平2−238558

| ⑤1Int.Cl.[5] | 識別記号 | 庁内整理番号 | ⑬公開　平成2年(1990)9月20日 |
|---|---|---|---|
| G 06 F 15/16 | 420 S | 6745−5B | |
| 9/445 | | | |
| 13/00 | 305 A | 8840−5B | |
| | | 7361−5B　G 06 F 9/06 | 420 J |

審査請求　未請求　請求項の数 1（全10頁）

⑤4発明の名称　並列計算機のブート方式

⑳特　願　平1−60060
㉒出　願　平1(1989)3月13日

⑦2発明者　清水　俊幸　神奈川県川崎市中原区上小田中1015番地　富士通株式会社内
⑦2発明者　石畑　宏明　神奈川県川崎市中原区上小田中1015番地　富士通株式会社内
⑦1出願人　富士通株式会社　神奈川県川崎市中原区上小田中1015番地
⑦4代理人　弁理士　井島　藤治　外1名

明　細　書

1．発明の名称

並列計算機のブート方式

2．特許請求の範囲

1個のホスト計算機（1）と複数個のプロセッサエレメント（4）とがバス（3）を介して接続された分散メモリ型並列計算機において、

各プロセッサエレメント（4）内に、

CPU（4a）と、

バス（3）と接続された通信ポート（4b）と、

プロセッサエレメントのリセット時にCPU（4a）から出力されるアドレスをデコードして通信ポート（4b）をアクセスするアドレスデコーダ（4c）とを具備し、

全てのプロセッサエレメント（4）が通信ポート（4b）をアクセスしたことをホスト計算機（1）側で検知したら、ホスト計算機（1）から各プロセッサエレメント（4）に対してブートプログラムを送出し、

各プロセッサエレメント（4）側では、通信ポート（4b）から入力されるブートプログラムをCPU（4a）により順次実行するように構成したことを特徴とする並列計算機のブート方式。

3．発明の詳細な説明

〔概要〕

1個のホスト計算機と複数個のプロセッサエレメントとがバスを介して接続された分散メモリ型並列計算機のブート方式に関し、

分散メモリ型並列計算機のプロセッサエレメントの数を増やせるという利点を十分に生かすことができるようにすることを目的とし、

各プロセッサエレメント内に、CPUと、バスと接続された通信ポートと、プロセッサエレメントのリセット時にCPUから出力されるアドレスをデコードして通信ポートをアクセスするアドレスデコーダとを具備し、全てのプロセッサエレメントが通信ポートをアクセスしたことをホスト計算機側で検知したら、ホスト計算機から各プロセッサエレメントに対してブートプログラムを送出し、各プロセッサエレメント側では、通信ポート

から入力されるブートプログラムをＣＰＵにより順次実行するように構成する。

［産業上の利用分野］

本発明は１個のホスト計算機と複数個のプロセッサエレメントとがバスを介して接続された分散メモリ型並列計算機のブート方式に関する。

近年、コンピュータシステムの高速化が要求されている。高速化の一つの実現法として、並列計算機が用いられる。ここで、並列計算機とは、プログラムを実行する計算要素（プロセッサエレメント；Ｐｒｏｃｅｓｓｏｒ　Ｅｌｅｍｅｎｔ、以下略してＰＥと記す）を複数個結合して一つの計算機を構成したものである。この種の並列計算機には、大きく分けて２つの実現方法が考えられる。一つは複数のＰＥで大きなメモリを共有する共有メモリ型並列計算機であり、もう一つはＰＥ毎に独立したメモリをもつ分散メモリ型並列計算機である。後者の分散メモリ型並列計算機は、ＰＥの数を大きくすることが可能であるという特徴をも



っている。ところが、一つのＰＥのハード量が大きくなるとこの特徴を生かすことができなくなる。このため、ＰＥのハードウェア量は小さく抑える必要がある。また、ＰＥの数が増加するに伴い、効率的なブート方法が要求されている。

［従来の技術］

第７図は、従来の分散メモリ型並列計算機の構成ブロック図である。１個のホスト計算機１と複数個のＰＥ２とがバス３を介して接続されている。第８図は各ＰＥの内部構成例（従来）を示す図である。ＰＥは、図に示すようにバス３と接続された通信ポート２ａ，ＲＡＭ２ｂ，ＣＰＵ２ｃ，ＲＯＭ２ｄ及びこれらを接続する内部バス２ｅより構成されている。ＲＯＭ２ｄ内にはブートアップ（ＩＰＬ；イニシャル・プログラムロード）用のプログラムが格納されている。

このように構成された分散メモリ型並列計算機のブートアップ（ＩＰＬ）は、ＰＥ内に用意されたＲＯＭ２ｄに格納されたプログラムによって行



われる。第９図は、従来のブートアップの手順を示すフローチャートである。先ず、ユーザがホスト計算機１を初期化する（Ｓ１）。その後、ホスト計算機１はＩＰＬを開始する（Ｓ２）。次に、ホスト計算機１はＩＰＬの一つの手順としてＰＥ２を初期化する（Ｓ３）。

各ＰＥ２はＲＯＭ２ｄ内に格納されているＩＰＬプログラムの実行を開始する（Ｓ４）。次に、ホスト計算機１はＰＥ２に対してＯＳ等をバス３を介して送信し、各ＰＥ２はＩＰＬの実行の過程で、ＯＳ等をホスト計算機１からバス３を介して受信する（Ｓ５）。そして、各ＰＥ２はＩＰＬを終了し、動作を開始し、ホスト計算機１はＩＰＬを終了し、動作を開始する（Ｓ６）。ここで、動作とは本来の並列処理動作をいう。

［発明が解決しようとする課題］

従来の方式では、各ＰＥ毎にブートプログラムを書込んだＲＯＭを用意し、そのプログラムによりブートアップ（ＩＰＬ）を行っていた。しかし



ながら、この構成をとるとＲＯＭの周辺等を含め、ある程度のハードウェアが必要となる。また、ＰＥの数だけＲＯＭ等を用意せねばならず、システム作製時のコストアップ及び動作時の信頼性の低下にもつながる可能性がある。以上により、従来システムではＰＥの台数を増やせるという利点を十分に生かすことができなかった。

本発明はこのような課題に鑑みてなされたものであって、分散メモリ型並列計算機のＰＥの数を増やせるという利点を十分に生かすことができる並列計算機のブート方式を提供することを目的としている。

［課題を解決するための手段］

第１図は本発明方式の原理ブロック図である。第７図と同一のものは、同一の符号を付して示す。図において、１はホスト計算機、３はバス、４はバスに接続された複数個のＰＥである。各ＰＥ４内には、ＣＰＵ４ａと、バス３と接続された通信ポート４ｂと、ＰＥのリセット時にＣＰＵ４ａか

ら出力されるアドレスをデコードして通信ポート４ｂをアクセスするアドレスデコーダ４ｃと、ＲＡＭ４ｄより構成されている。図では１つのＰＥについてその内部構成を示しているが、他のＰＥについても同様である。

［作用］

リセット時には、アドレスデコーダ４ｃはＣＰＵ４ａから出力されるアドレスデータをデコードして通信ポート４ｂをアクセスするようにする。そして、全てのＰＥ４が通信ポート４ｂにアクセスしたことをホスト計算機１側で検知したら、ホスト計算機１から各ＰＥ４に対してブートプログラムを送出し、各ＰＥ４側では、通信ポート４ｂから入力されるブートプログラムをＣＰＵ４ａにより順次実行するようにする。このような構成とすることにより、ＩＰＬ時のプログラムは各ＰＥ４に対してホスト計算機１から提供されるので、各ＰＥ４内にＩＰＬプログラムを格納したＲＯＭが不要となる。従って、本発明方式によれば分散



メモリ型並列計算機のＰＥの数を増やせるという利点を十分に生かすことができる。

［実施例］

以下、図面を参照して本発明の実施例を従来例と対比しつつ詳細に説明する。

本発明はＰＥのＣＰＵから見えるアドレス空間のデコードの方法を工夫することにより、ＲＯＭを必要としない方式としたものである。第２図はアドレス空間を示す図であり、（イ）は従来のアドレス空間を、（ロ）は本発明によるアドレス空間をそれぞれ示している。ここでは、次の仮定をしている。先ずＰＥのＣＰＵはホスト計算機から初期化（リセット）されると、アドレス０００００（＃は１６進を示す）から命令を取出し、実行を開始する。バスからのデータは、ポート（アドレスＦ０００＃）を読むことにより受取る。アドレスは全て１６進であり、図に示す値は例示である。

従来のアドレス空間は、（イ）に示すようにアド



レス０００００＃～２０００＃はＲＯＭに割当てられており、ここにＩＰＬプログラムが格納されていた。後の２０００＃～Ｆ０００＃まではＲＡＭ領域とＰＯＲＴ（ポート）領域が適宜割当てられていた。そして、ＰＥが初期化されると、ＰＥ内のＣＰＵはアドレス０００００＃から２０００＃の間に置かれたＲＯＭに格納されているＩＰＬによって動作に必要なＯＳ等（これらはホスト計算機により作成される）をポートから読出し、ＲＡＭに書込んでいく。

これに対し、本発明の場合には（ロ）に示すようにアドレス０００００＃から２０００＃までをポートのアドレスとしている。従って、ＰＥが初期化されてＰＥ内のＣＰＵが０００００＃からアドレスを出力すると、第１図に示したアドレスデコーダがこのアドレスをデコードしてポートアドレスに変換し、通信ポートをアクセスするようにする。この間に、ホスト計算機からＩＰＬのプログラムを各ＰＥに対して送出し、各ＰＥでは通信ポートを経由してＣＰＵにそのプログラムを与え、ＩＰ



Ｌを実行させるのである。従って、本発明によればＩＰＬプログラムを格納したＲＯＭは必要ないことになる。

次に、ホスト計算機がＰＥに送るデータとＰＥのＣＰＵが実行する命令の関係を更に詳細に説明する。ここでは、ＰＥのＣＰＵが実行する命令を以下のように定義する。

ＳＴ　ＡＤＤＲ　；アドレス（ＡＤＤＲ）にレジスタの値を書込む

ＬＤ　ＡＤＤＲ　；アドレスの（ＡＤＤＲ）の値をレジスタに読込む

また、ホスト計算機がＰＥに送るＯＳのデータ列をＯＳ０，ＯＳ１，…ＯＳＺ（ＯＳＺが最終データ，データの個数は１００＃と仮定）と書き表すものとすると、従来方式によりホスト計算機がＰＥに送るデータとＰＥのＣＰＵが実行する命令列は、第３図に示すようなものとなる。時刻ｔ１からｔ２までの間がホスト計算機から各ＰＥにＯＳを送信しているシーケンスである。

第４図はホスト計算機がＰＥに送るデータとＰ

ＥのＣＰＵが実行する命令例（本発明）を示す図である。従来例では、第３図に示すようにホスト計算機からはＯＳ命令のみが与えられているだけであったが、第４図の本発明の場合にはホスト計算機はＯＳ命令のみならずＬＤ　Ｆ０００ｈなる命令とＳＴ　２０００ｈなる命令を送っている。これら命令は、従来方式では内蔵のＲＯＭから与えられていたものである。ＰＥ側ではこのＬＤ命令が通信ポートから受取られるとＣＰＵの命令として実行される。つまり、ＰＥ側ではＣＰＵから出力されるアドレス００００ｈ～２０００ｈを全て通信ポートのアドレスＦ０００ｈに変換し、通信ポートから入力されるデータをＣＰＵが実行すべき命令として取り込み実行を進めていくものである。

第４図において、時刻$t_1$から$t_2$の範囲がホスト計算機から各ＰＥにＯＳを送信しているシーケンスである。前述したように、ＰＥのＣＰＵが実行する命令もホスト計算機からＰＥに送り出されている。このことは言い換えれば、従来ＲＯＭ



を付して示す。図では、ＰＥを１個しか示していないが、実際にはバス３に複数個接続されている。ホスト計算機１は、ＣＰＵ１ａ，メモリ１ｂ及びバス３を介してＰＥ４との接続制御を行うインターフェイス部１ｃより構成されている。このインターフェイス部１ｃには、バス３が接続される他に制御線５が接続されている。この制御線５は各ＰＥ４とも接続されている。ＰＥ４において、４ｅはバス３を介してホスト計算機１との接続制御を行うインターフェイス部、４ｆはＰＥ４内の内部バスである。第１図で示した通信ポート４ｂはインターフェイス部４ｅに含まれる。このように構成されたシステムの動作を説明すれば、以下のとおりである。

第６図は本発明によるブートシーケンスを示す図である。以下、このシーケンス図に沿って第５図に示すシステムの動作を説明する。先ず、インターフェイス部１ｃを介してホスト計算機１からＰＥへのリセット信号が出力される（①）。一方、ＰＥ４側では、インターフェイス部４ｅを介して



に格納していたＩＰＬをホスト計算機から送り出した命令で行っていることになる。従って、各ＰＥ内にＲＯＭを用意する必要がなくなったのである。このことが可能となったのは、前述したアドレスのデコードの工夫がポイントとである。更に説明する。ＰＥ内のＣＰＵは、初期化後アドレス０００００ｈから実行を始める。

そして、アドレス０００００ｈから命令を読込み、実行し、次にはアドレス０００１ｈから命令を読込み実行する。このように、アドレスを１つずつ更新しながら実行する。従来は、アドレス０００００ｈからのアドレス空間にＩＰＬを書込んだＲＯＭを置くことにより、ＩＰＬを実行していた。本発明では、この空間に通信ポートを割当て、ＣＰＵが初期化後、命令をアドレス０００００ｈから読込もうとすると、通信ポートのデータ、即ちホスト計算機からバスを介して送られてくる命令が読込まれることになる。

第５図は本発明の一実施例を示す構成ブロック図である。第１図と同一のものには、同一の符号



送られたきたリセット信号を受けて内部の状態をリセットして初期化する（（１））。リセットされると、ＣＰＵ４ａはアドレス０００００ｈから命令をフェッチして実行するようになっている。そこで、ＣＰＵ４ａはアドレス０００００ｈをアドレスデータとして出力する。このアドレスデータはアドレスデコーダ４ｃによってインターフェイス部４ｅ内の通信ポートをアクセスする信号に変換される。この結果、通信ポートがアクセスされる（（２））。しかしながら、制御線５を介してＡＣＫ信号（確認信号）がまだ有効になっていないのでそのままホールド状態となる（（３））。

一方、ホスト計算機側では、全てのＰＥ４が通信ポート４ｂをアクセスするのをインターフェイス部１ｃを介してＣＰＵ１ａにより監視している。そして、全てのＰＥが通信ポートをアクセスするのを待ってＰＥの第１命令を通信ポート４ｂに出力する（②）。また、それと同時に制御線５のＡＣＫ信号を有効にする（③）。

ＰＥ側ではＡＣＫ信号が有効になるまでホール

ドされていたが、ＡＣＫ信号が有効になったのを受けてＣＰＵ４ａが第１命令を読込み実行する（（４））。次に、ＣＰＵ４ａが第２の命令をフェッチするためのアドレス信号０００１ｈを出力すると、このデータは再度アドレステコーダ４ｃにより通信ポート４ｂをアクセスする信号に変換され、通信ポート４ｃをアクセスする（（５））。この時、ＡＣＫ信号は無効状態になっているので、ＡＣＫ信号が有効になるまでホールドされる（（６））。

ホスト計算機側では、全てのＰＥが通信ポートをアクセスするのを待って、ＰＥの次の命令を通信ポート４ｂに出力する（④）。それと同時に、制御線５のＡＣＫ信号を有効にする（⑤）。

ＰＥ側では、第２命令を通信ポート４ｂを介して読込み実行する（（７））。このようにしてＰＥ側では、ＣＰＵ４ａがフェッチする命令（命令アドレス）が１ＦＦＦｈを越えない間、（５），（６），（７）を繰り返す（（８））。一方、ＰＥ側ではブートシーケンスを終了するまで④，⑤



を繰返す（⑧）。ＯＳをホスト計算機から送る場合には、第４図で示したように、ホスト計算機は、ＰＥが実行すべき命令に合わせて、ＯＳのデータを送ればよい。

［発明の効果］

以上、詳細に説明したように、本発明によればＰＥがリセットされてからＰＥ内のＣＰＵが命令フェッチ用に出力するアドレスをデコードして通信ポートをアクセスする信号に変換してやり、ＩＰＬのための命令を通信ポート経由でホスト計算機から貰って実行する構成とすることにより、ＰＥ内のＲＯＭを不要とすることができる。従って、本発明によれば分散メモリ型並列計算機のＰＥの数を増やせるという利点を十分に生かすことができるようになる。

４．図面の簡単な説明

第１図は本発明方式の原理ブロック図、

第２図はアドレス空間を示す図、

第３図はホスト計算機がＰＥに送るデータとＰ



ＥのＣＰＵが実行する命令例（従来）を示す図、

第４図はホスト計算機がＰＥに送るデータとＰＥのＣＰＵが実行する命令例（本発明）を示す図、

第５図は本発明の一実施例を示す構成ブロック図、

第６図は本発明によるブートシーケンスを示す図、

第７図は従来の分散型並列計算機の構成ブロック図、

第８図は各ＰＥの内部構成例（従来）を示す図、

第９図は従来のブートアップの手順を示すフローチャートである。

第１図において、

１はホスト計算機、

３はバス、

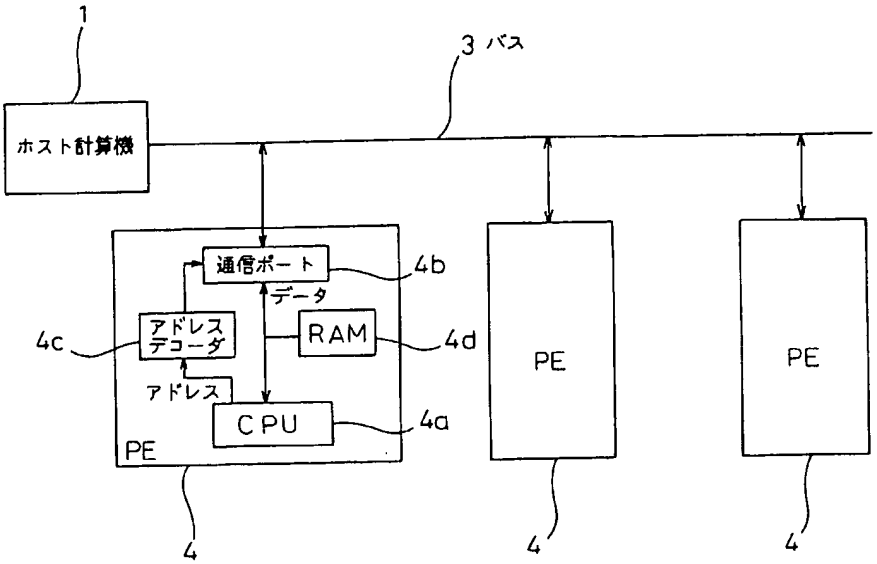
４はＰＥ、

４ａはＣＰＵ、

４ｂは通信ポート、

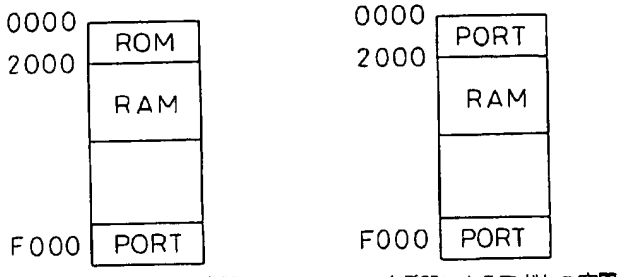
４ｃはアドレスデコーダ、

４ｄはＲＡＭである。




本発明方式の原理ブロック図

第 1 図

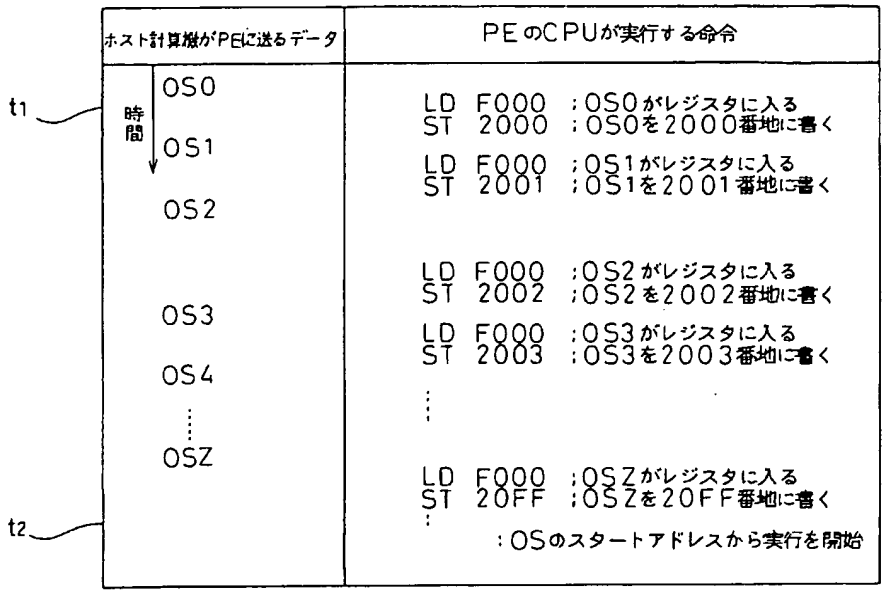



従来のアドレス空間
(イ)

本発明によるアドレス空間
(ロ)

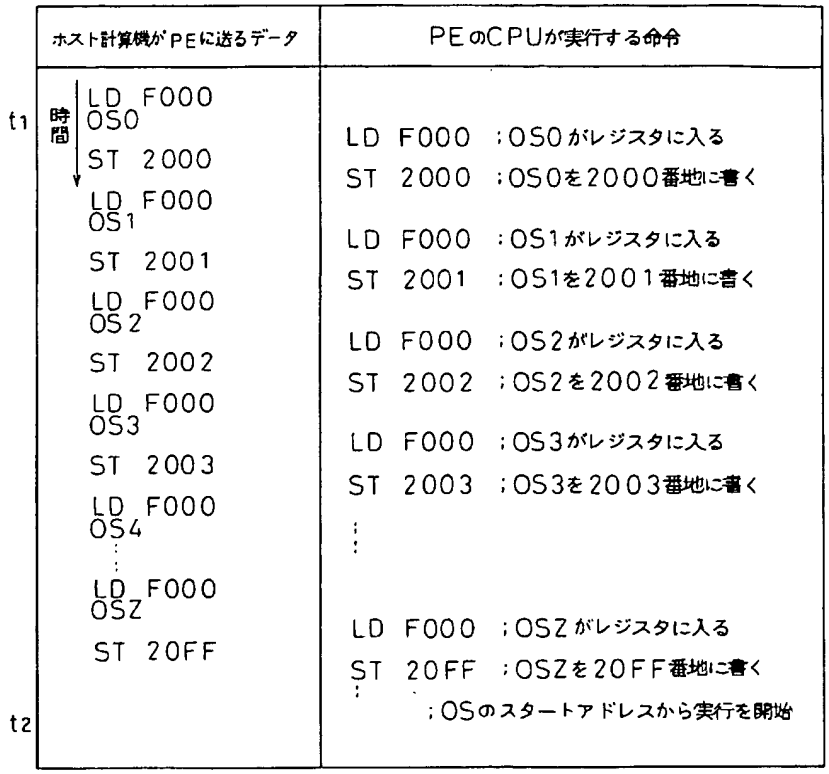
アドレス空間を示す図

第 2 図



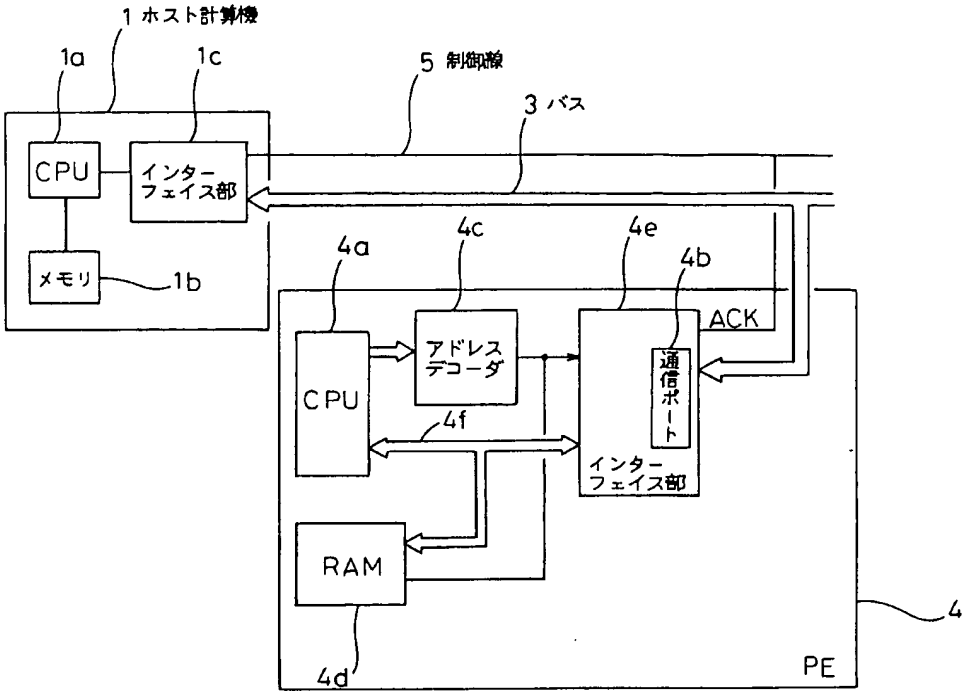

ホスト計算機がPEに送るデータとPEのCPUが実行する命令例(従来)を示す図

第 3 図




ホスト計算機がPEに送るデータとPEのCPUが実行する命令例(本発明)を示す図

第 4 図



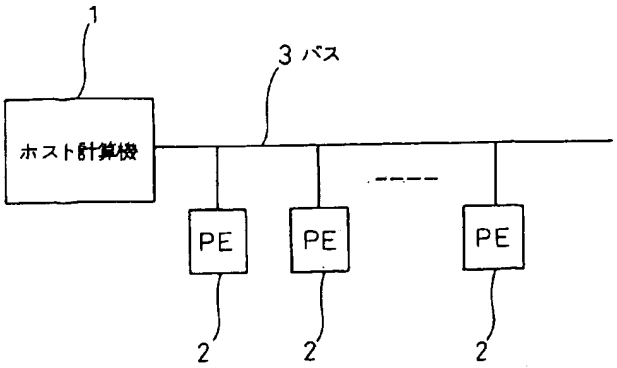

本発明の一実施例を示す構成ブロック図

第 5 図

| PE 側 | ホスト計算機側 |
|---|---|
| | ① PEをリセット |
| (1) リセット | |
| (2) 命令をアドレス0からフェッチ<br>(通信ポートをアクセス) | |
| (3) ACKが有効になるまでホールド | |
| | ② 全PEが通信ポートをアクセスするのを待って、PEの第1命令を通信ポートに出力する |
| | ③ ACKが有効になる |
| (4) 第1命令を読み込み実行 | |
| (5) 次の命令を次アドレスからフェッチ<br>(通信ポートをアクセス) | |
| (6) ACKが有効になるまでホールド | |
| | ④ 全PEが通信ポートをアクセスするのを待って、PEの次命令を通信ポートに出力する |
| | ⑤ ACKが有効になる |
| (7) 命令を読み込み実行 | |
| (8) 命令アドレスが1FFFを越えない間、<br>(5), (6), (7)を繰り返す | ⑥ ブートシーケンスが終了するまで、<br>④, ⑤を繰り返す |

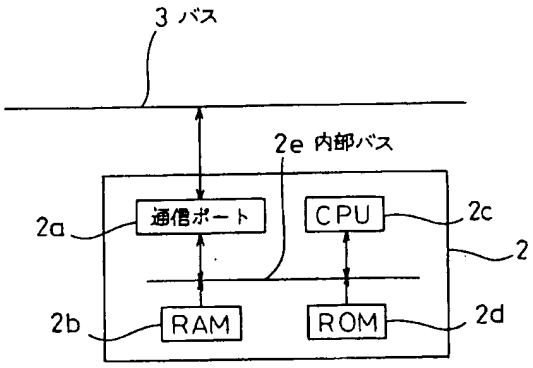
本発明によるブートシーケンスを示す図

第 6 図



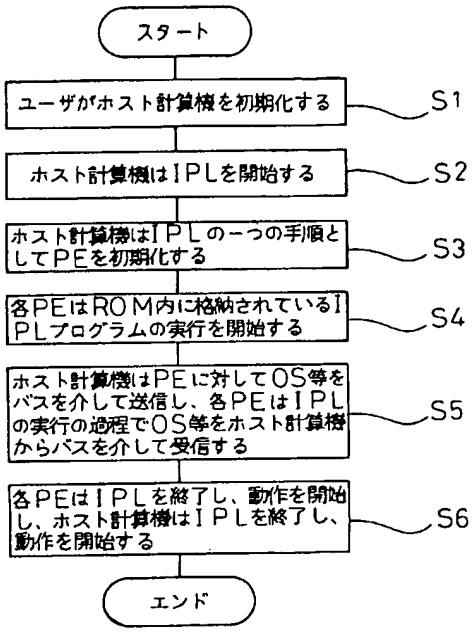

従来の分散メモリ型並列計算機の構成ブロック図

第 7 図




各PEの内部構成例従来を示す図

第 8 図




従来のブートアップの手順を示すフローチャート

第 9 図